Numerical Computing - CSCI 3656--001
Homework 7 10/22/21
Kai Handelman

Part 1-4 (Results with Theta = 1)

1.

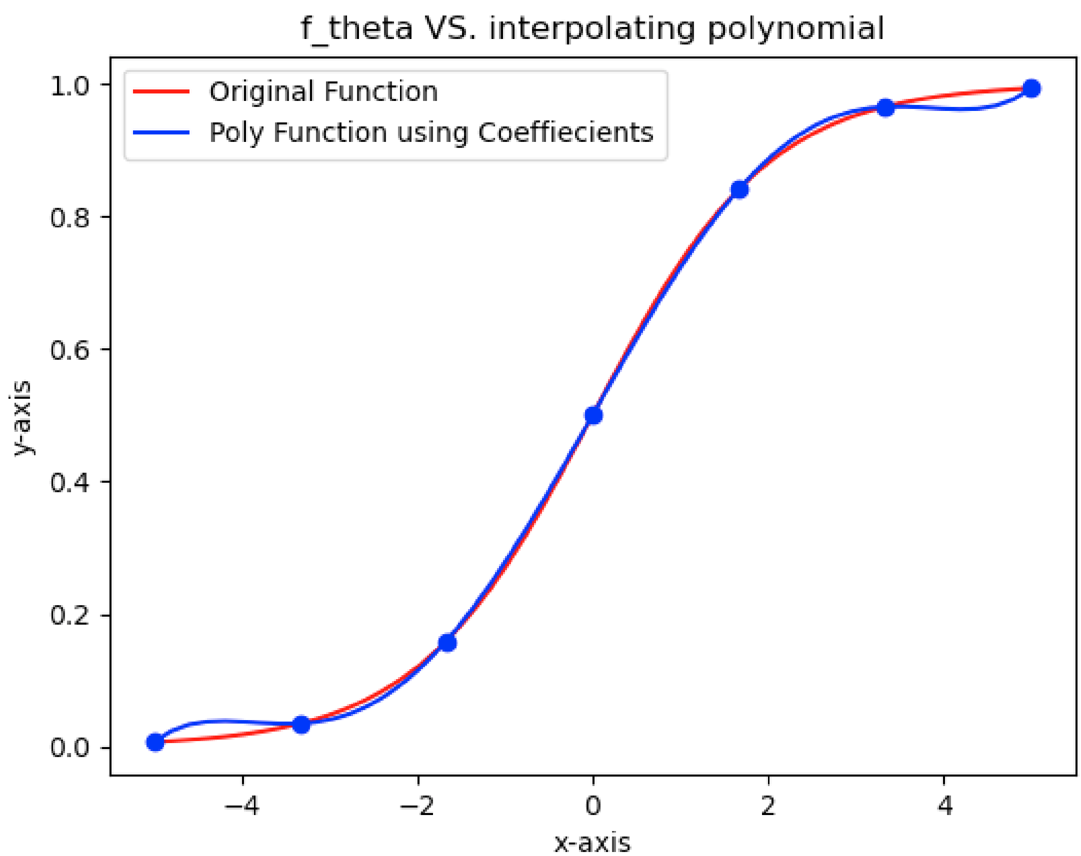
Running with theta value of: 1
Part 1: Input and Output of Function

x	f(x)
-5	0.00669285
-3.33333	0.0344452
-1.66667	0.158869
0	0.5
1.66667	0.841131
3.33333	0.965555
5	0.993307

2.

Part 2: Coeffiecients for Polynomial

x^n	C
0	0.5
1	0.233084
2	-1.07596e-16
3	-0.0108321
4	1.36244e-17
5	0.000218209
6	-3.69267e-19



This is a good approximation. Though not perfectly aligned, it is clear that for all 7 original points the interpolated function follows through. The biggest concern would be the ends where the Polynomial function starts to seem independent from the original function, however, we know that's a given if the sample points are equally distributed in an interval.

3.

```
The Mean for Part 3: 0.5000000000000001
The Standard Deviation for Part 3: 0.39019312637433506
```

4.

```
Maximum Error: 2.289837666290253
```

Part 5:

1.

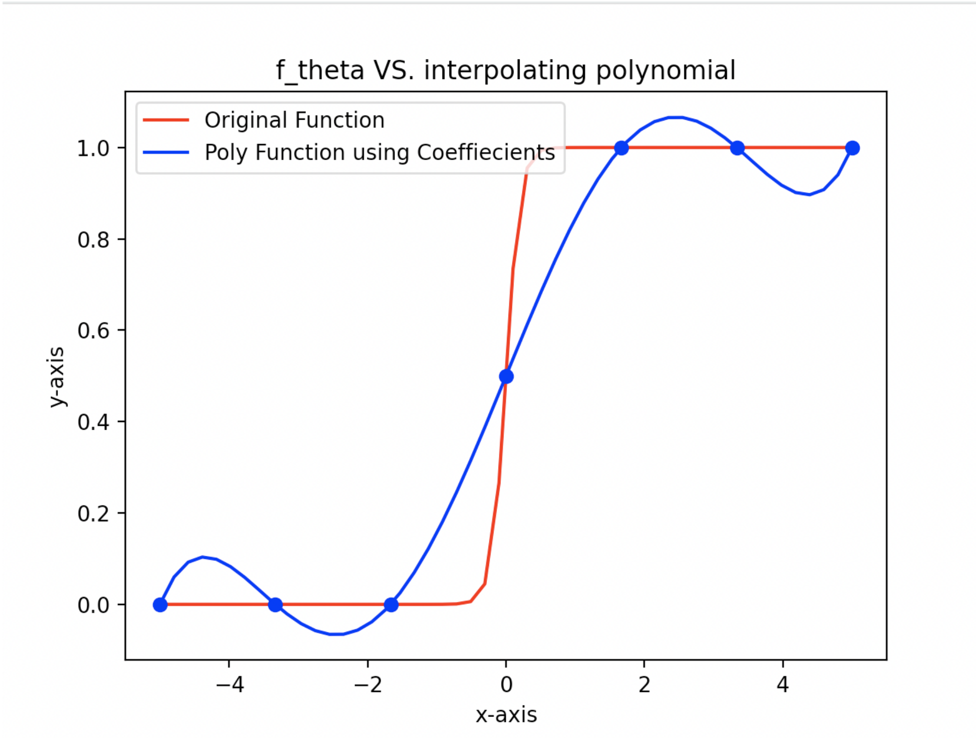
Running with theta value of: 10
Part 1: Input and Output of Function

x	f(x)
-5	1.92875e-22
-3.33333	3.33824e-15
-1.66667	5.77775e-08
0	0.5
1.66667	1
3.33333	1
5	1

2.

Part 2: Coeffieciants for Polynomial

x^n	C
0	0.5
1	0.37
2	-2.41339e-16
3	-0.027
4	3.2156e-17
5	0.000648
6	-9.00098e-19



Compared to when theta was 1, the approximation for theta = 10 fits a lot less with the actual function. This is due to how the steep-ness of with x approaches 0 is a lot more extreme in this case. This makes the interpolation never see the actual slope since only one point is in the range of the slope.

3.

```
The Mean for Part 3: 0.5
The Standard Deviation for Part 3: 0.4899989875597514
```

4.

```
Maximum Error: 6.310146146742918e+19
(base) kai@Kais-MacBook-Pro-2 Hw7 %
```

It is quite evident that the error increase by a huge margin when comparing theta = 1 vs theta = 10. This change can also be seen in the graphs where theta = 1 closely follows the original function why theta = 10 doesn't. As a result, we can conclude that the quality of the polynomial drastically dropped when theta when from 1 -> 10.

Code:

```
import numpy as np

import numpy.linalg as ln

from tabulate import tabulate

import matplotlib.pyplot as plt




def makePlot(fun,c,trueData):

    x = np.linspace(-5,5)

    f = fun(x)

    np.polyval

    # cF = lambda a: np.sum([np.power(a,i) * c[i] for i in range(len(c))])

    cF = np.polyval(np.flip(c),x)

    # print(x)

    plt.title("f_theta VS. interpolating polynomial")

    plt.xlabel("x-axis")

    plt.ylabel("y-axis")

    plt.plot(x,f,'r',label="Original Function")

    plt.plot(x,cF,'b',label="Poly Function using Coeffiecients")

    plt.legend(loc="upper left")

    for i in trueData:

        plt.plot(i[0],i[1], 'bo')

    plt.show()



def main(theta):
```

```
    print("Running with theta value of: {}".format(theta))

    #Part 1

    f = lambda x : 1/(1+np.exp(-theta*x))

    tData = np.linspace(-5,5,num=7)

    trueTable = [[t,f(t)] for t in tData]

    header = [["x","f(x)"]]

    print("Part 1: Input and Output of Function")

    print(tabulate(header+ trueTable,headers="firstrow",tablefmt='grid') + "\n")




    #Part 2

    a = [[np.power(trueTable[i][0],t) for t in range(len(trueTable))] for i in
range(len(trueTable))]

    b = [[t[1]]for t in trueTable]

    c = ln.solve(a,b)

    print("Part 2: Coeffiecients for Polynomial")

    header = [["x^n","C"]]

    print(tabulate(header+ [[i,c[i]]for i in
range(len(c))],headers="firstrow",tablefmt='grid') + "\n")

    # makePlot(f,c,trueTable)

    #TODO Make Table



    #Part 3

    nX = np.linspace(-5,5,num=101)

    nY = [[f(t)]for t in nX]

    print("The Mean for Part 3: {}".format(np.average(nY)))
```

```
    print("The Standard Deviation for Part 3: {}".format(np.std(nY)))



    #Part 4

    err = 0

    nCF = [np.polyval(np.flip(c),i) for i in nX]

    # cF = lambda a: np.polyval(np.flip(c),a)

    for i in range(len(nX)):

        temp = np.abs(nY[i][0] - nCF[i][0])/np.abs(nY[i][0])

        if temp > err:

            err = temp

    print("Maximum Error: {}".format(err))









main(1)

# print("\n\n\n")

#Part 5

main(10)
```